The built-in solver for EUROPA is a generic one, and therefore are not very efficient for some domains. There are three typical ways to make the planning more efficient.

1. Configure the built-in planner by changing PlannerConfig?.xml An introduction of the general idea is [here].

A general PlannerConfig?.xml looks like the following:

```
<Solver name="DefaultTestSolver">
  <FlawFilter component="HorizonFilter" policy="PartiallyContained"/>

  <ThreatManager defaultPriority="0">
    <FlawHandler component="StandardThreatHandler"/>
    <FlawFilter class-match="Reservoir"/>
    <FlawFilter class-match="Reusable"/>
  </ThreatManager>

  <OpenConditionManager defaultPriority="0">
    <FlawHandler component="StandardOpenConditionHandler"/>
  </OpenConditionManager>

  <UnboundVariableManager defaultPriority="0">
    <FlawFilter var-match="start"/>
    <FlawFilter var-match="end"/>
    <FlawFilter var-match="duration"/>
    <FlawFilter class-match="Resource" var-match="time"/>
    <FlawFilter class-match="Resource" var-match="quantity"/>
    <FlawFilter class-match="Reservoir" var-match="time"/>
    <FlawFilter class-match="Reservoir" var-match="quantity"/>
    <FlawFilter class-match="Reusable" var-match="quantity"/>
    <FlawFilter component="InfiniteDynamicFilter"/>
    <FlawHandler component="StandardVariableHandler"/>
  </UnboundVariableManager>
</Solver>
```

Basically, it contains three pairs of managers and handlers, one for each type of flaws. **defaultPriority** sets the default priority of each flaw in this type (0 the highest). A **FlawFilter?** says ignore a set of flaws that has the following properties. Other than filter a flaw (a filtered flaw will never be addressed by the planner, so that if that's part of the domain constraints, the planner might return a plan that conflicts with the filtered flaw), you might want to give a low priority to some type of flaws. The following line shows how this is done:

```
<FlawHandler component="StandardThreatHandler" predicate="medical_conference" order="late" priority="2"
```

Basically, this line says that when we use standard threat handler as our flaw handler, we set the priority to be 2 (less prioritied than priority 0 or priority 1), when the predicate happenes to be medical_conference. The **order** keyword says when we try values for this flaw, we try large values for timeline first (supposed to be late).

2. Write your own class of handlers. This can be done by writing a C++ class somewhere.

3. Write your own planner.